nature | methods

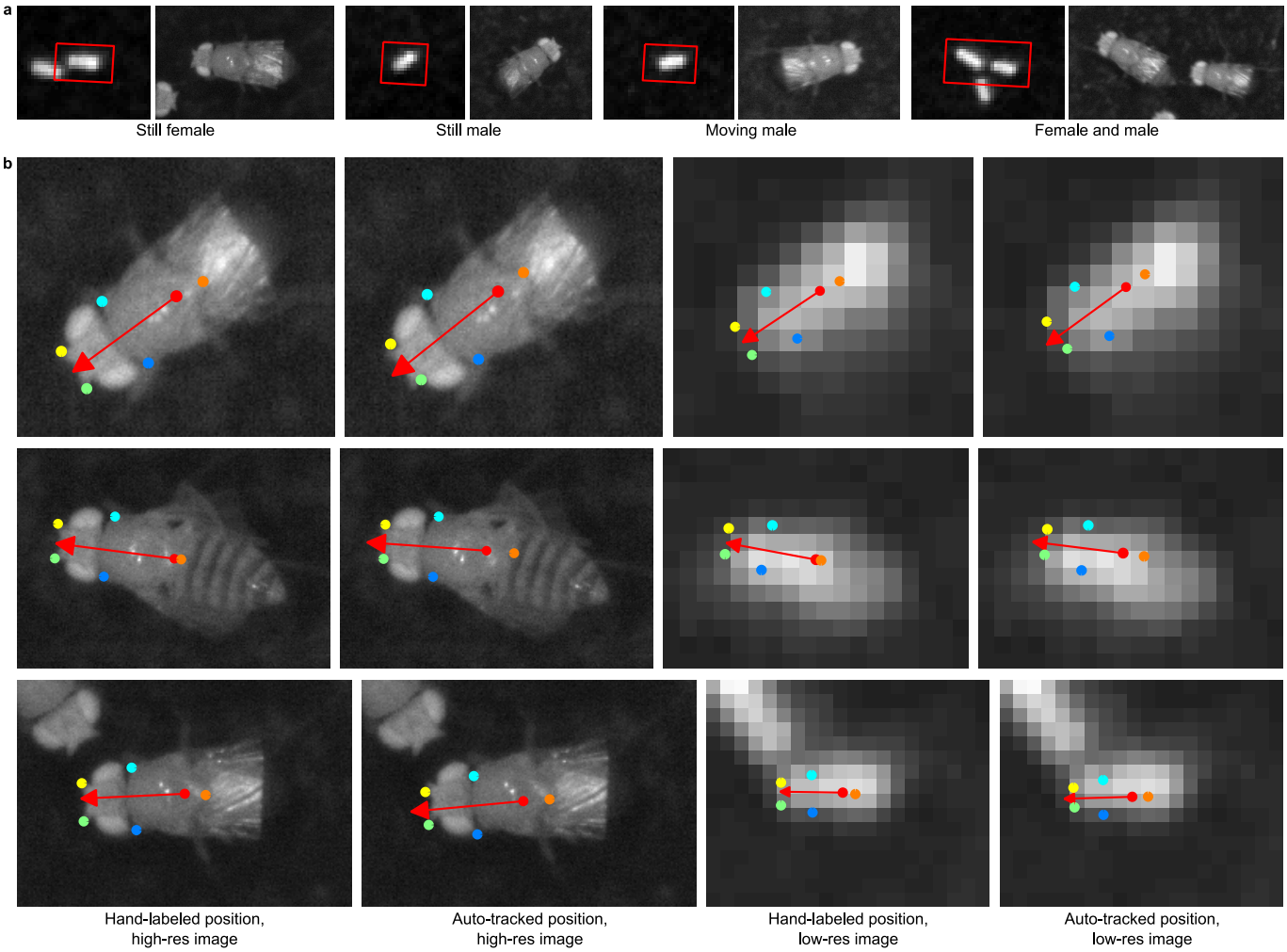
High-throughput ethomics in large groups of *Drosophila*

Kristin Branson, Alice A Robie, John Bender, Pietro Perona & Michael H Dickinson

Supplementary figures and text:

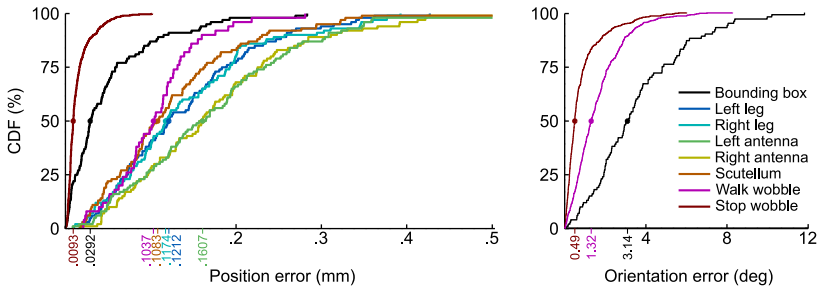
Supplementary Figure 1	llustration of groundtruthing using high-resolution video.
Supplementary Figure 2	Cumulative distribution of position errors.
Supplementary Figure 3	Extended behavioral vectors.
Supplementary Figure 4	Summary statistics of behavior.
Supplementary Table 1	Identity errors.
Supplementary Table 2	Parameters of each of the eight behaviors.
Supplementary Table 3	Position errors for close flies.
Supplementary Note	

Note: Supplementary Software and Supplementary Videos are available on the Nature Methods website.



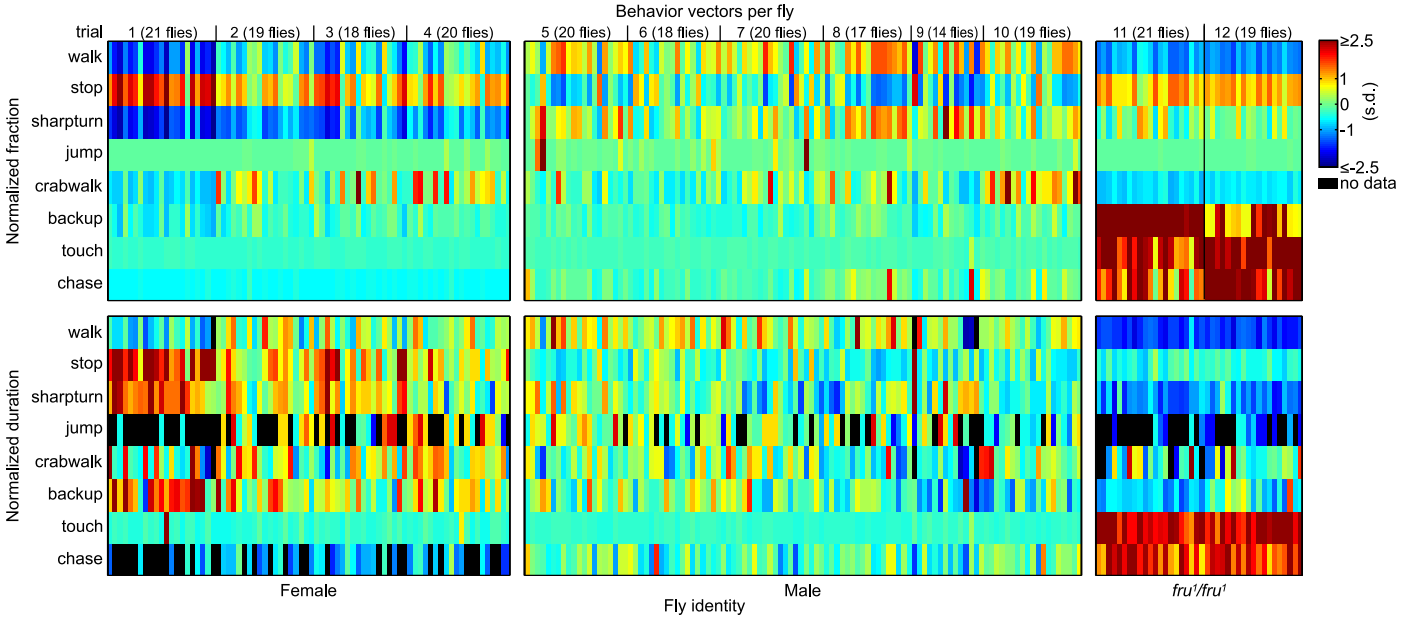
Supplementary Figure 1: Illustration of groundtruthing using high-resolution video. (**a**) Example images of the same flies captured by the low- and high-resolution cameras. The left image of each pair is from the low-res camera, the right image is from the high-res camera. The red box in the low-resolution images indicates the location of the adjacent high-resolution image. The last pair of frames shows a male and female fly near each other. (**b**) Examples of high resolution labels and corresponding low-resolution tracks. Each row shows the manually labeled high resolution image and the positions estimated by the tracker for a different example. The first column shows the true labels plotted on the high-resolution frame. The second column shows the projection of the positions estimated by the tracker on the high-resolution frame. The third column shows the projection of the true labels on the low-resolution frame. The last column shows the positions estimated by the tracker on the low-resolution frame. The last row shows touching flies. The red circle and arrow show the bounding box-based center position and orientation. The colored circles indicate the positions of the five parts.

Nature Methods: doi:10.1038/nmeth.1328



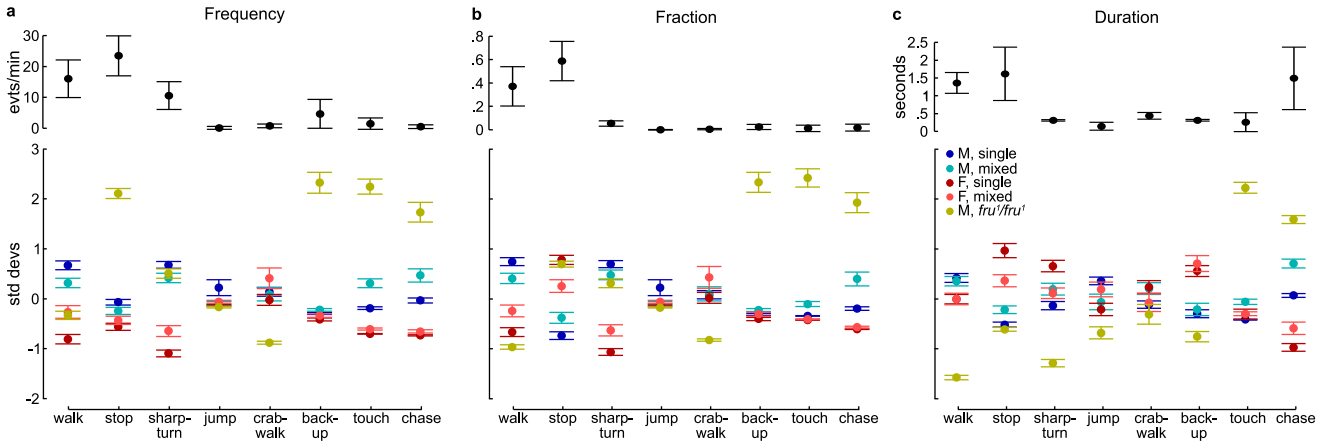
Supplementary Figure 2: Cumulative distribution of position errors. We plot the cumulative distribution of position errors for each of the methods of estimating error. On the left, we plot error in estimating point locations. On the right, we plot error in estimating the orientation of the fly. The line labeled 'Bounding box' corresponds to errors in estimating the center and orientation of the fly by manually drawing a bounding box on the high resolution images. The next five lines correspond to errors in estimating the positions of the parts manually labeled on the high resolution images. The 'Walk wobble' line corresponds to estimates of the center and orientation error computed by smoothing the velocity and orientation during walks. The 'Stop wobble' line corresponds to estimates of the center and orientation computed from the variance in position of a stopped fly. On the x-axes, the colored ticks mark the median values.

Nature Methods: doi:10.1038/nmeth.1328



Supplementary Figure 3: Extended behavioral vectors for each male, female, and fru^1/fru^1 fly (see **Fig. 3e** of the main text for details). In the top image, the color indicates the (normalized) fraction of time each fly performs each behavior. In the bottom image, the color indicates the (normalized) mean duration of bouts of each behavior for each fly. Mean duration cannot be computed for flies that do not perform a given behavior; this is indicated by black.

Nature Methods: doi:10.1038/nmeth.1328



Supplementary Figure 4: Summary statistics of behavior. For each fly and behavior, we computed (a) the frequency of onsets of the behavior (**Fig. 3d-f**), (b) fraction of time the fly performed the behavior (**Supplementary Fig. 3**, top), and (c) mean duration of sequences of the behavior (**Supplementary Fig. 3**, bottom). The mean and standard deviation of these statistics over the entire fly population is shown in the top row (black). In the bottom row (colored), we show the normalized (z-scored) mean and standard error for each of the five fly and trial types (male/wild type/single-sex, ..., female/wild type/mixed-sex, fru^1/fru^1. These plots show that many of these behavioral statistics are significantly different for different fly types.

Nature Methods: doi:10.1038/nmeth.1328

Supplementary Table 1: Identity errors. Number of each type of identity error for each video evaluated. Each row corresponds to a different video. The first two columns show the number of female and male flies in the video. The third column shows the number of frames in the video. The fourth column shows the number of times a pair of flies was close (note that this number can be greater than the number of frames in the video, as multiple pairs of flies may be close in a single frame). The next five columns show the number of errors of each type: *Swap* identity, *Lost* fly, *Spurious* track, *Merged* fly, and *Split* fly.

Fly Sex		N. Frames		N. Errors					Error Freq	
♀*	♂	Total	Close	Swap	Lost	Spurious	Merged	Split	Swaps/Close	Errors/Fly/Frame
10	0	6364	42	0	0	0	0	0	0	0
10	0	6118	67	1	0	0	0	0	1.5e-2	1.9e-05
20	0	6221	261	0	1	0	0	0	0	9.4e-06
20	0	6206	358	0	1	0	0	0	0	8.5e-06
50	0	6123	2135	0	2	0	0	0	0	7.3e-06
50	0	6587	4545	1	0	0	0	0	2.2e-4	7.1e-06
0	10	6106	268	0	0	0	0	0	0	0
0	10	6133	316	0	0	0	0	0	0	0
0	20	6169	1237	0	1	0	0	0	0	8.1e-06
0	20	6118	1029	0	1	0	0	0	0	8.8e-06
0	50	8696	9765	4	0	1	0	0	4.1e-4	1.2e-05
0	50	6088	4855	7	1	0	0	0	1.4e-3	3.0e-05
5	5	6268	499	0	0	0	0	0	0	0
5	5	7859	753	0	0	0	0	0	0	0
10	10	6121	2050	1	1	0	0	0	4.9e-3	1.9e-05
10	10	6104	1460	0	1	0	0	0	0	8.9e-06
25	25	6220	9843	1	3	0	0	0	1.0e-4	1.4e-05
25	25	6536	9902	1	1	1	0	0	1.0e-4	9.6e-06

*__Column Key__: ♀: number of female flies. ♂: number of male flies. *Tot*: *Tot*al number of frames in the video. *Clo*: Number of frames, pairs of flies in which the flies are *clo*se to each other. *Swa*:*Swa*p identity errors. *Los*: *Los*t fly errors. *Spu*: *Spu*rious track errors. *Mer*: *Mer*ged fly errors. *Spl*: *Spl*it track error occurs. *Swa*/*Clo*: Ratio of number of identity *swa*ps to frames in which the flies are *clo*se.

Nature Methods: doi:10.1038/nmeth.1328

Supplementary Table 2: Parameters for each of the 8 behaviors. Each row for a behavior corresponds to a different property that is computed at every time instant. The 'Property' column gives this property an abbreviated name; the exact definitions are described below the table. The four Property columns show the ranges the property is allowed to take for each of the four conditions enumerated in the Methods section. The 'Min Length' column shows the minimum length of the sequence. The 'Radius' column shows the definition of 'near' for property (2): frame t_1 is near frame t_2 if t_1 is within Rad seconds of t_2.

Behavior	Property	Parameter ranges				Min Length (s)	Radius (s)
		Per-frame (1)	Near frame (2)	Sum (3)	Mean (4)		
walking	speed (mm/s)	$[9.9, \infty)$	$[11.9, \infty)$	$[2.4, \infty)$	$[10.6, \infty)$	.25	.2
	$\lvert\dot{\theta}\rvert$ (°/.05s)	$[0, 31.3]$	$[0, 19.8]$		$[0, 5.2]$		
	acc (mm/(.05 s)^2)	$[0, 2.0]$	$[0, 0.8]$		$[0, 1.1]$		
	$\lvert smth(\theta)\rvert$ (°/.05s)	$[0, 12.9]$	$[0, 10.5]$				
stopped	speed (mm/s)	$[0, 4.8]$	$[0, 1.7]$	$[0, 2.6]$	$[0, 3.1]$	.35s	.1
	$\lvert\dot{\theta}\rvert$ (°/.05s)	$[0, 5.7]$	$[0, 2.3]$	$[0, 9.7]$	$[0, 3.1]$		
	acc (mm/(.05s)^2)	$[0, 1.00]$	$[0, 0.80]$		$[0, 0.88]$		
sharp turn	$\lvert\dot{\theta}\rvert$ (°/.05s)	$[4.0, \infty)$	$[6.6, \infty)$	$[27.6, \infty)$	$[9.3, \infty)$	.35	.1
	speed (mm/s)	$[0, 27.2]$	$[0, 9.2]$	$[0, 4.5]$	$[0, 13.6]$		
	acc (mm/(.05s)^2)	$[0, 1.4]$	$[0, 1.2]$		$[0, 1.2]$		
crabwalk	$\lvert\phi - \theta\rvert$ (°/.05s)	$[7.1, \infty)$	$[26.8, \infty)$	$[99.7, \infty)$	$[18.9, \infty)$	.5	.1
	$\lvert\dot{\theta}\rvert$ (°/.05s)	$[0, 21.6]$	$[0, 3.9]$	$[0, 151.8]$	$[0, 7.7]$		
	$\lvert smth(\theta)\rvert$ (°/.05s)	$[0, 16.9]$	$[0, 12.6]$		$[0, 6.7]$		
	$\perp$ tail vel (mm/s)			$[1.54, \infty)$	$[0.54, \infty)$		
backing up	fwd vel (mm/s)	$(-\infty, 0]$	$(-\infty, -0.50]$	$(-\infty, .08]$	$(-\infty, 0.50]$	.15	.15
chasing	min speed (mm/s)		$[9.9, \infty)$		$[8.5, \infty)$	.55	.55
	vel twd (mm/s)	$[-18.6, \infty)$	$[-6.5, \infty)$	$[-5.0, \infty)$	$[3.0, \infty)$		
	max speed (mm/s)		$[3.3, \infty)$		$[14.3, \infty)$		
	dist btn (mm)	$[0, 26.1]$	$[0, 12.1]$		$[0, 7.8]$		
	speed apt (mm/s)	$[0, 41.2]$	$[0, 37.7]$		$[0, 14.7]$		
	angle btn (°)	$[0, 79.0]$	$[0, 62.9]$		$[0, 61.7]$		
	θ diff (°)	$[0, 180.5]$	$[0, 120.3]$		$[0, 98.4]$		
	ϕ (°)	$[0, 173.7]$	$[0, 108.2]$		$[0, 40.9]$		
jumping	speed (mm/s)		$[48.1, \infty)$			0	.05
touching	dist btn (mm)		$[0, 2.5]$			.25	.05
	ang btn (°)		[0,0]				

- **Property key**:
- **speed**: Magnitude of the vector between the fly's centroids in frames $t-1$ to t, normalized by the frame rate. The integral of speed (column 3) is the total distance traveled during the segment, measured in mm.
- $|\dot{\theta}|$: Absolute value of change in orientation from frame $t-1$ to t, normalized by the frame rate. The integral of $|\dot{\theta}|$ is the absolute value of the total change in orientation during the segment, measured in degrees.
- **acc**: Absolute change in velocity between frames $t-1$ and $t+1$, measured in mm per frame^2 $= (0.05s)^2$
- $|\dot{smth}(\theta)|$: Absolute change in the smoothed orientation from frame $t-1$ to t, normalized by frame rate = $0.05s/frame$.
- $|\phi - \theta|$: Absolute difference between orientation at frame t and velocity direction from frames t to $t+1$.
- $\perp$ **tail vel**: Change in position of tail of fly from frames t to $t+1$, projected onto the direction orthoganal to the fly's orientation in frame t.
- **fwd vel**: Dot product of orientation vector at frame t and vector connecting centroids in frames t to $t+1$, normalized by frame rate.
- **min speed**: Minimum speed of either fly involved in the chase.
- **vel twd**: Dot product of direction of chased fly from chasing fly and vector connecting centroids of chasing fly in frames t to $t+1$, normalized by frame.
- **max speed**: Maximum speed of either fly involved in the chase.
- **dist btn**: Distance between the head of the chasing fly and any point on the boundary ellipse of the chased fly
- **speed apt** Magnitude of the difference vector between the velocity vectors of each fly from frames $t-1$ to t, normalized by frame rate.
- **angle btn** Absolute difference between angle from chasing fly to chased fly and chasing fly's orientation.
- θ **diff** Absolute difference between flies' orientations.
- ϕ **diff** Absolute difference between flies' velocity directions.
- **ang btn**: Minimum absolute angle between touching fly's orientation and direction to any point on other fly.

Nature Methods: doi:10.1038/nmeth.1328

Supplementary Table 3: Position errors for close flies.

Type	Median	Std
Center	0.046 mm	0.090 mm
Orientation	2.8°	2.0°
Left Antenna	0.18 mm	0.09 mm
Right Antenna	0.19 mm	0.09 mm
Left Leg	0.21 mm	0.09 mm
Right Leg	0.21 mm	0.09 mm
Scutellum	0.14 mm	0.10 mm

Each row of this table corresponds to a different error measure computed from the high-resolution groundtruth samples in which another fly was near the labeled fly. The Type column describes the measure, the Median column lists the median error in the estimate and the Std column lists the standard deviation of the error of the estimate. The first two rows show the error for the center position and orientation esimates, computed from the bounding box labels. The last five rows show the error estimates for the labeled parts.

Nature Methods: doi:10.1038/nmeth.1328

Supplementary Note

1 Behavior Definitions

(Details related to Behavior Definitions section of Methods)

In this section, we describe how we chose the quantitative definitions of the behaviors analyzed in the Results section. We also provide a table with the properties and precise thresholds we used (**Supplementary Table 2**). We provide code for automatically detecting these behaviors online (`http://dickinson.caltech.edu/ctrax`).

Six of our behaviors (walk, stop, sharp turn, crabwalk, backup, and jump) involve basic locomotor actions. The majority of the time, the flies either walked at a relatively constant velocity or stopped in place. The next-most common behavior was 'sharp turn', in which a fly makes a large and rapid change in orientation. Sharp turns usually occurred in response to the arena wall or another fly. Most commonly, the fly pivoted around its hind end, though occasionally it pivoted around its center or front end. These front-end pivots primarily occurred during 'encounters' with another fly (see below). Other locomotor classifications included 'crabwalks', in which the fly walked with a substantial sideways component, and 'backups', in which the flies' translational velocity was negative. Both crabwalks and backups were often exhibited as a result of encounters with other flies. Jumps consisted of rapid translations within the arena, often as a consequence of encounters with other flies. Our two remaining behaviors (touch and chase) related to social interactions between flies. A touch occurred when the head of one fly came in contact with another fly. Chases were cases in which one fly (always a male) followed another across the arena.

For 6/8 behaviors (walking, stopping, making sharp turns, jumping, crabwalking, and chasing), it was time-consuming and non-intuitive to manually examine the parameter space for the precise set of parameters that best fit our intuition about the behavior definitions. Thus, we chose the parameters by manually labeling a number of trajectories. These labeled trajectories are the training examples used to learn the parameter intervals. The advantage of this learning-based approach is that to obtain an automatic behavior classifier a behavior expert can simply specify segmented training trajectories for a behavior of interest. He/she does not need to write any code. We chose the structure of our classifier so that the training examples are converted into 'characteristic' parameter ranges for each behavior. We thus end up with a clear and concise definition of each behavior that is open to inspection and criticism and makes experiments easily reproducible.

Given the segmented example trajectories, we then manually chose which properties the classifier would depend on, and whether it would be bounded from above, below, or both for each of the four conditions described in the Methods section. We used a genetic algorithm [1] to find the ranges that produced segmentations closest to the labeled trajectories. Details of the algorithm are given in Supplementary Note Sec. 1.1. Our cost function for comparing a proposed segmentation and the groundtruth segmentation is

$$cost = (\#\text{ spurious detections}) + (\#\text{ missed detections}) + 0.2(\#\text{ mislabeled frames})$$

Spurious detections are detected sequences that don't match any labeled sequence, and missed detections are labeled sequences that don't match any detected sequence. We say that two segments match if the proposed segmentation can be made identical to the groundtruth by relabeling at most 5 frames = .25 seconds at the beginning, end, or middle. We made the definition of matching tolerant to small errors because the exact frame a behavior begins on is subject to interpretation. The exact details of the scoring function are shown in Algorithm 1.

Algorithm 1 Score an automatically detected behavior labeling $x_{1:T}$ against the groundtruth $y_{1:T}$.

Input: Proposed labeling $x_{1:T}$ and groundtruth labeling $y_{1:T}$.
Output: Measure of the number of errors in the labeling $cost$.

$cost \leftarrow 0$
for t_1, t_2: $y_{t_1:t_2} \neq x_{t_1:t_2}, x_{t_1-1} = y_{t_1-1}, x_{t_2+1} = y_{t_2+1}$ **do**
  $length \leftarrow t_2 - t_1 + 1$
  if $length > 5$ **then**
    $cost \leftarrow cost + 1$
  end if
  $cost \leftarrow cost + length/5$
end for

We observed holistically that the segmentations produced by the learned classifiers match our intuitive definitions of the behaviors. In **Supplementary Video 5**, we show examples of behaviors that were labeled manually, and of behaviors that were detected automatically.

1.1 Cross-Entropy Method

To find the behavior definition parameters that minimize the cost criterion described above, we use the cross-entropy method [1]. The cross-entropy method is initialized with hyperparameters that describe the prior distribution on the parameters (see Supplementary Note Sec. 1.2). It then generates a set of N joint samples of all parameters[1] according to this prior. The cost criterion is evaluated for each sample. The M

[1] We set $N = 100$ samples

'elite' samples with the highest scores are chosen.[2] The hyperparameters defining the prior are reset to those which maximize the likelihood of these elite samples. These steps are iterated until there is no decrease in the highest cost elite sample over a span of d iterations[3].

1.2 Cross-Entropy Initialization

For all parameters, we assume independent Gaussian prior distributions. To set the initial mean and variances of these prior distributions, for each positive training sequence, the system computes the most aggressive parameter values that would result in the sequence being classified as positive. The prior distribution mean is initialized to the average of the largest and smallest aggressive parameter values over all training examples. The prior standard deviation is initialized to one quarter of the difference between the largest and smallest aggressive parameter values.

2 Flies

(Details related to Flies section of Methods)

Unless noted, all experiments were carried out on adult *Drosophila melanogaster* selected from a laboratory population that was derived from 200 wild-caught isofemale female lines. Flies were maintained on 16:8 light:dark cycle and all experiment were conducted during the evening activity peak. Approximately 24 hours prior to each experimental trial, we collected between 20 and 50 flies (2 day-old) from culture bottles and anesthetized them using a cold plate cooled to 2°C. While they were anesthetized we clipped both wings of each fly to approximately 1/2 their normal length so that they could not fly out of the arena. After cold anesthetization the flies are allowed to recover overnight in food vials and 6 hours prior to experiments were transferred to vials with damp paper for wet starvation.

3 Apparatus

(Details related to Apparatus section of Methods)

The walking arena is depicted in **Figure 1** of the main paper. The temperature of the 6-mm-thick, 245-mm-diameter aluminum platform was actively controlled by an array of 4 circular thermoelectric modules (TE Technologies) bolted to the underside. The distal heating side of each module was attached to a temperature exchanger through which water was actively circulated and cooled by a radiator. The thermoelectric modules were driven to a set point of 25°C in parallel by a PID controller (FerroTec) with feedback from a thermistor also mounted to the bottom of the platform. The surface temperature varied by less than 1°C, as measured by an infrared non-contact thermometer (OmegaScope). The surface of the platform was lightly sandblasted and painted with ultra flat black spray paint (Krylon), which is minimally reflective in the near infrared. The surface was cleaned with detergent and distilled water between trials. The thermal barrier consisted of a 0.2 mm thick strip of galvanized steel wrapped around the platform, flush to the top surface. A rope heater (OmegaLux) was wrapped tightly around the steel strip and powered by a variable AC transformer (Staco), which heated the barrier in open loop. The thermal barrier was separated from the platform by a layer of 2 mm thick strip of neoprene so that the edge of the arena itself was not heated. To provide a visual stimulus, we placed a paper cylinder around the platform. The paper was printed with a random pattern of squares with 50% filling probability. Each square subtended 5° of azimuth as seen from the center of the arena. The cylinder was backlit by an array of eight halogen lights, powered by DC current to avoid visual stimulus flicker, creating a luminance of 450 lux at the center of arena floor. The arena was illuminated with near-infrared LEDs (Dale Wheat) mounted in the same horizontal plane as the camera above the platform. We used a 1280x1024-pixel, firewire camera (Basler A622f) equipped with an 8 mm lens (PENTAX). This imaging solution gaves us a resolution of 4 pix/mm, i.e. flies were 8 pixels long. An infrared pass filter was placed in front of the camera to block stray light from the arena. Images were recorded onto hard disk at 20 frames per second by a computer using the Motmot Python camera interface package[2]. Each group of experimental flies was allowed to walk into the arena from a vial through a hole in the floor that was plugged once all the flies had entered (larger groups used for ground-truthing were simply dumped from a vial into the center of arena and allowed to settle for 1 minute).

While our software was developed in conjunction with this set up, it is adaptable to other arrangements with similar characteristics. In particular, we have successfully tracked flies in another arena that does not require wing clipping, as shown in **Supplementary Videos 6** and **7**. In this alternate setup, we use a Sigma-Coat coated glass ceiling to keep the flies from walking or flying from the arena. The floor of the translucent arena is illuminated with IR light to create a backlit image of the flies.

4 Tracking Algorithm Details

(Details related to Tracking Algorithm section of Methods)

In this section, we provide details of our tracking algorithm. First, we describe the preprocessing steps, in which the tracker learns what the arena image looks like without flies in it (Supplementary Note Sec. 4.1), what shapes a fly can take (Supplementary Note Sec. 4.2), and what regions of the image flies are in (Supplementary Note Sec. 4.3). Then, we provide details of how the positions of flies in the current video

[2] We set $M = .25N = 25$

[3] We use $d = 5$

frame are estimated (Supplementary Note Sec. 4.4). Next, we describe how the observed fly positions are assigned identities by matching them with the positions predicted from the previous frames (Supplementary Note Sec. 4.5). We then describe how tracks are modified in hindsight so that track births and deaths are avoided (Supplementary Note Sec. 4.6). Finally, we describe the post-processing step to resolve the head-tail orientation ambiguity (Supplementary Note Sec. 4.7).

4.1 Background Modeling

(Details related to Detection section of Methods)

We model the background pixel intensities at each location in the image as independent Gaussian distributions. Instead of fitting each Gaussian using the maximum likelihood estimates (the sample mean and standard deviation) we use the more robust median and median absolute deviation. That is, the tracker estimates the center of the Gaussian $\mu(\mathbf{p})$ at a given pixel location $\mathbf{p}$ as the median pixel intensity of the sampled frames at that location:

$$\mu(\mathbf{p}) = \underset{\{t=0,\Delta,2\Delta,\ldots,T)\}}{\text{median}} I_t(\mathbf{p}),$$

where I_t is the video frame at time t, T is the number of frames in the video, and Δ is the interval between sampled frames.[4] At each frame, the tracker also computes the absolute difference between the observed pixel value $I_t(\mathbf{p})$ and the median $\mu(\mathbf{p})$. The tracker estimates the standard deviation from the median such absolute difference:

$$\sigma(\mathbf{p}) = c \underset{\{t=0,\Delta,2\Delta,\ldots,T)\}}{\text{median}} |I_t(\mathbf{p}) - \mu(\mathbf{p})|$$

where the constant c ensures that the correct fraction of the data is within one standard deviation:

$$c = 1/(\sqrt{2}\text{erf}^{-1}(.5)) \approx 1.4826.$$

4.2 Shape Modeling

(Details related to Detection section of Methods)

As discussed in detail in Supplementary Note Sec. 4.4.3, the tracker decides whether to merge, split, or delete connected components based on the expected image area of a fly. The shape parameters are computed automatically. To compute the female shape parameters, in 50 frames evenly spaced through each all-female video, the tracker detected all connected components and computed the mean and standard deviation of their areas. The maximum and minimum area bounds are set to the mean plus and minus three standard deviations. The same computation is performed for the male parameters using the all-male videos. The mixed arena parameters were set as the extrema of the single-sex parameters (since females are larger than males, the minimum area was set to the minimum area for males and the maximum area was set to the maximum area for females).[5]

Currently, we only use the fly's area to model its shape, but in future work we plan to use the length of the major and minor axes and the eccentricities of the ellipses as well.

4.3 Arena Detection

(Details related to Detection section of Methods)

The tracker automatically detects the circular arena floor by fitting a large circle to edges in the background image using the Hough circle transform [3]. All pixels outside of the arena floor are labeled as background. This step is necessary because the wall of the arena is extremely reflective. As the flies are restricted from walking on the wall by the heat barrier, most foreground pixels on the wall are due to reflections of flies on the arena floor. In addition, in future experiments we plan to use dynamic LED panels to affect the flies' behavior, as in [4]. Thus, we would like to ignore all foreground detections not on the floor of the arena. In future work, we plan to extend the tracker interface to allow arena shapes other than the circle.

4.4 Observation Detection

(Details related to Detection section of Methods)

[4] We chose $\Delta = \lfloor T/200 \rfloor$ in all our experiments.

[5] In all our experiments, we used the same fly area parameters. The minimum area was 16.77 px^2 = 1.0488 mm^2 for all-male arenas, 21.445 px^2 = 1.3155 mm^2 for all-female arenas, and 16.78 px^2 = 1.0488 mm^2 for mixed arenas. The maximum area was 63.19 px^2 = 3.9494 mm^2 for all-male arenas, 76.945 px^2 = 4.8091 mm^2 for all-female arenas, and 76.945 px^2 = 4.8091 mm^2 for mixed arenas. The mean area was the average of these extreme values.

Here, we overview the steps involved in estimating the positions of flies in the current frame. To understand the choices made, it is beneficial to view observation detection in terms of probabilistic modeling and inference. The general goal in detection is to find the positions of the flies that best explains the current video frame. More formally, we would like to find the positions of the flies of maximum density given the current video frame. Let $\mathbf{x}_i = (x, y, \theta, a, b)^\top$ be the ellipse position for the ith fly detected (the x- and y-coordinates of the centroid, the orientation, the semi-major axis length, and the semi-minor axis length), $\mathcal{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ be a set of N fly positions, and I be the current video frame. We would like to find $\mathcal{X}$ that maximizes

$$p(\mathcal{X}|I) \propto p(\mathcal{X})p(I|\mathcal{X}).$$

We assume that the prior density on fly positions is independent for each fly:

$$p(\mathcal{X}) = \prod_{i=1}^{N} p(\mathbf{x}_i).$$

Our prior on the position of a single fly $p(\mathbf{x}_i)$ is the model of the fly's area:

$$p(\mathbf{x}_i) \propto \exp[-|\pi a_i b_i - \mu_{area}|/\sigma_{area}].$$

To compute the likelihood of the video frame I given the fly positions $\mathcal{X}$, $p(I|\mathcal{X})$, we construct the foreground/background labels predicted for each pixel location given $\mathcal{X}$. The label $L_{ij}(\mathcal{X})$ predicted at pixel location (i, j) is labeled foreground if it is part of a fly, that is, if it is inside the ellipse for some fly in $\mathcal{X}$, and background if it is not. We assume that the intensity of each pixel in the video frame is independent given its label, thus the likelihood factors as

$$p(I|\mathcal{X}) = \prod_{(i,j)} p(I_{ij}|L_{ij}(\mathcal{X}))$$

where $p(I_{ij}|background)$ is the Gaussian background model described in Supplementary Note Sec. 4.1 and $p(I_{ij}]|foreground)$ is uniform.

Finding the optimal fly positions is difficult because $p(\mathcal{X}|I)$ has many local maxima and is non-smooth, and $\mathcal{X}$ has a discrete component (the number of flies). We therefore use a sequence of heuristics to try to optimize the criterion efficiently. First, each pixel is classified as foreground or background (Supplementary Note Sec. 4.4.1). Each connected component of foreground pixels usually corresponds to exactly one fly. Thus, initially one ellipse is fit to each of these connected components (Supplementary Note Sec. 4.4.2). To identify connected components which do not correspond to exactly one fly, the tracker finds ellipse fits that have a low density according to the prior $p(\mathbf{x}_i)$. For these connected components, the number of ellipses fit to the connected component is iteratively increased or decreased to optimize the criterion (Supplementary Note Sec. 4.4.3).

4.4.1 Background Subtraction

Classifying a pixel location as foreground (belonging to a fly) or background (not belonging to any fly) is referred to as background subtraction [5]. To do this classification, the tracker thresholds the likelihood of the observed pixel intensities in the current frame given the background model described in Supplementary Note Sec. 4.1. Pixels with high likelihood are labeled background and pixels with low likelihood are labeled foreground. As we use a Gaussian model, it is equivalent to threshold the absolute difference from the mean normalized by the standard deviation:

$$l(\mathbf{p}) = \begin{cases} \text{foreground} & |I(\mathbf{p}) - \mu(\mathbf{p})|/\sigma(\mathbf{p}) > thresh \\ \text{background} & \text{otherwise} \end{cases}$$

The tracking software allows us to specify the special case that flies are always brighter than the background (or vice-versa). In this case, the signed difference $(I(\mathbf{p}) - \mu(\mathbf{p}))/\sigma(\mathbf{p})$ is thresholded.

To improve robustness to noise, when thresholding foreground from background pixels, we use a hysteresis approach. The tracker thresholds the difference at a low threshold $thresh_{small}$. If no pixel in the connected component has a difference larger than a larger threshold $thresh_{large}$, thenall labels in this connected component are flipped to background.[6]

4.4.2 Ellipse Fitting

To fit an ellipse to a connected component of foreground pixels, the tracker fits a 2-D Gaussian to the locations of the foreground pixels in the connected component. Given the parameters of the best-fitting Gaussian, the parameters of the ellipse can be computed. Consider all the pixel locations within an ellipse with semi-major axis length a, semi-minor axis length b, and orientation θ. The sample mean of the pixel locations will be close to the ellipse center. The sample covariance of the pixel locations within this ellipse will be close to

$$\Sigma = R^\top \begin{pmatrix} a/2 & 0 \\ 0 & b/2 \end{pmatrix}^2 R,$$

[6] In all our experiments, we set $thresh_{small} = 10$ and $thresh_{large} = 20$.

where

$$R = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}.$$

Thus, given the covariance matrix Σ, the semi-major and semi-minor axis lengths and orientation can be computed by finding the eigen-decomposition of $\Sigma = U^\top DU$. The axis lengths are twice the square root of the eigenvalues, and the orientation can be computed as the arctangent of two of the entries of U:

$$a = 2\sqrt{D_{11}}, \qquad b = 2\sqrt{D_{22}}, \qquad \theta = atan(U_{12}, U_{11}).$$

Instead of just computing the sample mean and covariance of all the pixels in the connected component, the tracker computes a weighted mean and covariance, where the weight of a pixel is proportional to its distance from the background model. Let $\{\mathbf{p}_1, ..., \mathbf{p}_n\}$ be the locations of the pixels in a given connected component. Let the weight of pixel i be the normalized distance of the pixel intensity from the background image:

$$w_i = |I(\mathbf{p}_i) - \mu(\mathbf{p}_i)|/\sigma(\mathbf{p}_i).$$

Then the sample mean and covariance are computed as

$$Z = \sum_i w_i$$

$$\mu = \frac{1}{Z}\sum_i w_i \mathbf{p}_i$$

$$\Sigma = \frac{1}{Z}\sum_i w_i (\mathbf{p}_i - \mu)(\mathbf{p}_i - \mu)^\top$$

Using the weighted mean and variance improves our accuracy in two ways. First, it improves robustness to less than perfect threshold parameters, and allows us to use a single threshold throughout the image, despite lighting differences in different regions. In dimmer parts of the arena, low threshold is needed. In brighter parts, using this low threshold results in pixels at the edge of the fly being classified as foreground. These pixels are actually background, but contain some foreground light from blurring effects. While they are classified as foreground, they have a relatively small weight in the estimate of the ellipse. Second, using the weighted mean and variance improves the subpixel accuracy of our estimates.

4.4.3 Splitting and Merging Connected Components

If the prior density $p(\mathbf{x}_i)$ of an ellipse fit $\mathbf{x}_i$ for a given connected component is small, then this connected component may actually correspond to multiple flies, part of a fly, or a spurious detection. Recall that the prior density is currently based only on the area of the ellipse. If the prior is small because the area of the ellipse is too large, then the tracker determines if fitting multiple flies to the connected component will increase the score $p(\mathcal{X}|I)$. If the prior density is small because the area of the ellipse is too small, then the tracker determines if any of the following will increase the score: (1) lowering the foreground threshold to increase the area, (2) merging the ellipse with nearby ellipses, or (3) deleting the ellipse.

Splitting Large Components. Because of image blur and the legs of the flies, the pixels surrounding the fly will actually be a blur of foreground and background. Often, these pixels will be classified as foreground but have a higher background likelihood than pixels in the interior of the fly. We observed that relabeling these pixels as background often resulted in large connected components consisting of multiple flies being split into multiple connected components. Thus, the tracker first determines whether raising the foreground threshold for the connected component results in multiple, reasonably sized components.[7] If it does not, then the Expectation-Maximization algorithm is used to fit a mixture of K Gaussians to the weighted pixel locations in the large connected component [6]. This approximately finds a fit of K ellipses (that do not overlap too much) to the weighted pixel locations in the large connected component. More specifically, a mixture of K Gaussians is the distribution

$$p(\mathbf{p}) = \sum_{k=1}^{K} \pi_k \mathrm{G}(\mathbf{p}; \mu_k, \Sigma_k),$$

where π_k is the relative weight of component k, μ_k and Σ_k are the parameters of the k^{th} Gaussian, and $\mathrm{G}(\mathbf{p}; \mu, \Sigma)$ is the density of the Gaussian with parameters μ and Σ at pixel location $\mathbf{p}$.

The tracker determines the number of components K to split the large component into based on the shape prior $\prod_{k=1}^{K} p(\mathbf{x}_k)$ of the ellipses fit. We empirically observed that this prior did not usually have multiple maxima, thus the tracker greedily chooses the number of components. It iteratively increases the number of components it splits the large component into, stopping when there is a decrease in the prior.

Fixing Small Connected Components. If a connected component has a small area, the tracker first determines if the ellipse fit is small because the foreground threshold is too high in that region of the image. Note that, in our setup, the ideal threshold varied with location in the image,

[7] The tracker iteratively tries increasing the threshold from $thresh_{small}$ to the maximum normalized distance in the connected component for 20 iterations, stopping if it has successfully split the large connected component.

as the lighting varied with location in the arena. The tracker decreases the foreground threshold for pixels near the connected component and refit the ellipse.[8] If this does not sufficiently increase the component's area, then the tracker determines whether the connected component corresponds to part of a fly. For each nearby component, the tracker determines whether merging the small component with this nearby component results in a small decrease in the image likelihood $p(I|\mathcal{X})$[9] and a large increase in the prior $p(\mathcal{X})$. If the area could not be increased sufficiently in either of these two ways, and the area is smaller than a given threshold[10], then the connected component is deleted.

4.5 Identity Matching

(Details related to Identity Assignment section of Methods)

To match the tracks of flies in previous frames with the observed fly positions in the current frame, we use a constant velocity model. This model allows us to predict the positions of the flies in the current frame given their positions in the previous frames. Then, the predicted positions are matched to the observed positions minimizing this error.

4.5.1 Constant Velocity Prediction

In a constant velocity model, we assume that the velocity of the fly from frame $t-1$ to the current frame t is the same as the previous velocity from frame $t-2$ to $t-1$. If $\mathbf{x}_{t-1}$ is the position of the fly in frame $t-1$ and $\mathbf{x}_{t-2}$ is the position of the fly in frame $t-2$, then the constant velocity prediction is that the fly will be at

$$\mathbf{x}^{pred} = \mathbf{x}_{t-1} + (\mathbf{x}_{t-1} - \mathbf{x}_{t-2})$$

in frame t. We use a constant velocity model for the center of the fly, and a dampened velocity model for the orientation of the fly, as we observed that the orientation velocity is somewhat noisy. In this dampened velocity model, we predict the current orientation velocity to be some fraction of the previous orientation velocity:

$$\theta^{pred} = \theta_{t-1} + \lambda_{dampen}(\theta_{t-1} - \theta_{t-2})_{(-\pi/2,\pi/2]}$$

where λ_{dampen} is a constant between 0 and 1 and the angle difference wraps around at $-\pi/2$ and $\pi/2$.[11]

The penalty for matching the predicted fly position $\mathbf{x}^{pred} = (x^{pred}, y^{pred}, \theta^{pred})$ and the observed fly position $\mathbf{x}^{obs} = (x^{obs}, y^{obs}, \theta^{obs})$ is computed as

$$err(\mathbf{x}^{pred}, \mathbf{x}^{obs}) = (x^{pred} - x^{obs})^2 + (y^{pred} - y^{obs})^2 + w_{orient}(\theta^{pred} - \theta^{obs})^2_{(-\pi/2,\pi/2]}$$

where w_{orient} is the weight of the orientation error relative to the position error, and the angle difference is computed on the interval $(-\pi/2, \pi/2]$.[12] In future work, we plan to explore more complex, accurate models of fly motion.

4.5.2 Finding the Optimal Matching

The error function $err(\mathbf{x}_u^{pred}, \mathbf{x}_v^{obs})$ defines the cost of assigning identity u to the v^{th} detection. If we simply assign to the v^{th} detection the identity u with the minimum error, then we may end up with multiple flies in the current frame with the same identity, and no flies with other identities. Here, we describe the algorithm for computing the best one-to-one matching of identity to observation. By one-to-one matching, we mean that every fly identity is assigned to exactly one observation, and every observation is assigned to exactly one identity.

In fact, the situation is a little more complicated than this, as there is the potential that a fly may escape the arena, a new fly may enter the arena, or there may be an error in detection. For simplicity, let us ignore this possibility for now; we will return to it later.

In this simplified setup, let N be the number of flies observed in the current frame (and therefore the number of identities to assign). We can represent a matching as a set of pairings of identities and observations: $A = \{(i_1, o_1), ..., (i_N, o_N)\}$, where $i_i \in \{1, ..., N\}$ is the identity assigned to observation $o_i \in \{1, ..., N\}$. The penalty for a matching is the sum of the penalties of each individual assignment, as described in the previous section:

$$err(A) = \sum_{j=1}^{N} err(\mathbf{x}_{i_j}^{pred}, \mathbf{x}_{o_j}^{obs}).$$

We require that no identity be assigned to multiple observations: $i_j \neq i_k \forall j \neq k$, and that no observation be assigned multiple identities: $o_j \neq o_k \forall j \neq k$. We also require that every identity and every observation be in some assignment. Our goal is to find the lowest error legal matching.

[8] The threshold was decreased to 1 standard deviation.

[9] Corresponding to at most 40 px^2 = 2.5 mm^2 increase in normalized distance^2

[10] 5 px^2 = .3125 mm^2

[11] We used $\lambda_{dampen} = 0.5$.

[12] We used $w_{orient} = 100px^2/rad^2 = 6.25mm^2/rad^2 = 0.044mm/deg.$

Nature Methods: doi:10.1038/nmeth.1328

Let us now expand the definition of a matching and its error to allow some detections or observations not to be matched. In general, we would prefer an observation to be assigned an existing identity, as we are assuming that track deaths and births are unlikely. However, if forcing an assignment requires that the fly accelerate a huge amount, then we would prefer to assign a new identity to the fly. Similarly, if the error for assigning an identity to some observation is too large, we would prefer to let this fly track die.

As the number of identities to assign may be different than the number of observations, let M be the number of identities to assign. We represent observation o being assigned a new identity as an assignment (i, o) to a dummy identity $i \in \{M+1, ..., M+N\}$. We represent an identity i not being assigned to any observation as an assignment (i, o) for a dummy observation $o \in \{N+1, ..., N+M\}$. We now represent a matching as the set of $M+N$ pairs $A = \{(i_1, o_1), ...(i_{M+N}, o_{M+N})\}$ where $i_j \in \{1, ..., M+N\}$ and $o_j \in \{1, ..., M+N\}$. Pairs (i, j) where both the identity and observation are dummy variables ($i > M$ and $j > N$) do not symbolize anything.

Let max_{err} be the maximum amount we expect a fly can accelerate from one frame to the next. We used the fairly large value 100 (corresponds to the fly jumping 25 mm if there is no change in orientation) for this threshold, as flies occasionally jump and it is rarer that a fly leave or enter the arena. We modify the definition of a matching error to include assignment of an observation to a new identity and assignment of an identity to no observation:

$$err(i,o) = \begin{cases} err(\mathbf{x}_i^{pred}, \mathbf{x}_o^{obs}) & i \leq M, o \leq N \\ max_{err} & i > M, o \leq N \\ max_{err} & i \leq M, o > N \\ 0 & i > M, o > N \end{cases}$$

Finding the minimum error perfect matching A is an instance of the square assignment problem, a.k.a. the minimum weight perfect bipartite matching problem [7]. It can be solved quickly using a number of algorithms. We use the Hungarian method [7] to solve this problem (`http://mit.edu/harold/Public/hungarian.tgz`).

4.6 Hindsight

(Details related to Hindsight section of Methods)

In the previous steps of the tracking algorithm, a number of decisions were made without incorporating all available information. In particular, the observed fly positions in the current frame are fixed without considering the positions of the flies in previous and future frames. Second, our computation of the error of matching an identity with an observation only incorporates information from the previous frames, not the future frames. Third, the error of an assignment is the sum of the errors for each pair of matches independently. In this step, the tracker determines if a track death or birth in the current frame can be avoided by fixing potential errors in the previous ΔT frames[13]. The tracker determines if tracks were previously merged, split, lost, or the result of spurious detections, resulting in the birth or death of a track in the current frame.

In a merged detection, two flies are tracked as a single fly for a short sequence of frames[14]. A merged detection will exhibit itself as the death of track i_1 in frame t_1 (i.e. identity i_1 will not be assigned to any observations in frame t_1) and the birth of track i_2 in frame $t_2 > t_1$ (the corresponding observation in frame t_2 will not be assigned an existing identity). In addition, there will be some track i_3 that can be split into two tracks from frames t_1 to $t_2 - 1$ with low penalty[15]. Finally, the errors of matching the split track to i_1 in frame t_1 and i_2 in frame t_2 will be small[16]. Given the birth of a track i_2 in frame t_2, the tracker therefore first searches the recent history for such tracks i_1 and i_3. The tracker then splits i_3 in frames t_1 to $t_2 - 1$ and connected the split tracks to i_1 and i_2.

In a split detection, a single fly is split into two detections for a short sequence of frames[17]. A split detection will exhibit itself as the birth of track i_1 in frame t_1 and the death of i_1 soon after in frame t_2. In addition, there will be another track i_2 that i_1 can be merged with in frames t_1 through $t_2 - 1$ with low penalty[18]. Given the death of a track i_1 in frame t_2, the tracker therefore first searches the recent history for such a track i_2. The tracker then merges track i_1 with i_2 in frames t_1 to $t_2 - 1$, and replace identity i_2 with i_1.

In a lost detection, a fly is not detected for a short sequence of frames[19]. A lost detection will exhibit itself as the death of track i_1 in frame t_1 followed by the birth of track i_2 in frame t_2 soon after. In addition, the predicted position of fly i_1 in frame t_2 will be close to the observed position of i_2 in frame t_2[20]. Given the birth of fly track i_2 in frame t_2, if the tracker cannot connect i_2 to previous tracks by splitting, it will then search for such a track i_1. It then connects these two tracks. It replaces identity i_2 with i_1, and set the positions of i_1 in frames $t_1 + 1$ through $t_2 - 1$ by linear interpolation.

In a spurious detection, the tracker detects a fly for a short sequence of frames where there is no fly[21]. In this case, a fly i will be born in

[13] In our experiments, ΔT = 50 frames = 2.5 seconds
[14] $\leq$ 50 frames = 2.5 s
[15] $\leq$ 40 px^2 = 2.5 mm^2
[16] $\leq 20 \approx$ 5 mm^2
[17] $\leq$ 50 frames = 2.5 s
[18] $\leq$ 40 px^2 = 2.5 mm^2
[19] $\leq$ 50 frames = 2.5 s
[20] within 100 px $\approx$ 25 mm
[21] $\leq$ 50 frames = 2.5 s

frame t_1 and die soon after in frame t_2. Given the death of fly track i in frame t_2, if it cannot connect it to other tracks by merging, it deletes this track if its lifespan is short enough.

4.7 Resolving Orientation Ambiguity

(Details related to Orientation Ambiguity section of Methods)

Because of the low resolution of the video, it is difficult to visually distinguish the head from the tail of the fly. The orientation of the fly computed online is only known modulo π; it is not known whether the orientation is θ_t or $\theta_t + \pi$. To resolve this ambiguity, we make the following assumptions. First, when a fly is walking quickly, its head will point forward, i.e. the orientation of its body and the direction of the fly's velocity will approximately match. Second, we assume that the fly's orientation does not change much from one frame to the next. In this section, we define a criterion that combines these two assumptions, then show how to minimize this criterion.

Let θ_t be the orientation of the fly at frame t, ϕ_t be the velocity direction at frame t, v_t be the speed of the fly at time t, and T be the number of frames in the entire video. We are searching for the binary values $s_t \in \{0, 1\}$ which indicate whether we will add π to the orientation: $\theta'_t = \theta_t + \pi s_t$. We find the sequence of states $\mathbf{s}_{1:T}$ that minimizes the following criterion:

$$\begin{aligned} J(\mathbf{s}_{1:T}) &= \sum_{t=1}^{T} [J^1(s_t) + J^2(s_t, s_{t-1})] \\ J^1(s_t) &= w(v_t) |(\theta_t + \pi s_t - \phi_t)_{(-\pi,\pi]}| \\ J^2(s_t, s_{t-1}) &= (1 - w(v_t)) |(\theta_t + \pi s_t - \theta_{t-1} - \pi s_{t-1})_{(-\pi,\pi]}| \end{aligned}$$

The first term $J^1(s_t)$ penalizes the orientation $\theta_t + \pi s_t$ differing from the velocity direction ϕ_t. When the fly is sitting still, it may still have a non-zero velocity. However, the direction of this velocity is not related to the orientation of the fly. Thus, we weight this error term proportional to the magnitude of the velocity of the fly. We use the weight function

$$w(v) = \min\{w_{max}, \lambda v^2\},$$

where λ and w_{max} are constants[22]. The second term $J^2(s_t, s_{t-1})$ penalizes the orientation at frame t, $\theta_t + \pi s_t$, differing from the orientation at frame $t-1$, $\theta_{t-1} + \pi s_{t-1}$.

The global optimum of this criterion can be found efficiently (space and time $O(T)$) using dynamic programming [8]. In order to do so, $J_t(s_{1:t})$ needs to be rewritten recursively as

$$\begin{aligned} [C_t(s_t), p_t(s_t)] &\triangleq \min_{s_{1:t-1}} J_t(s_{1:t-1}, s_t) \\ &= \min_{s_{t-1}} [\min_{s_{1:t-2}} J_{t-1}(s_{1:t-2}, s_{t-1}) + \\ &\quad J^1(s_t) + J^2(s_t, s_{t-1})] \\ &= \min_{s_{t-1}} [C_{t-1}(s_{t-1}) + J^1(s_t) + J^2(s_t, s_{t-1})]. \end{aligned}$$

$C_t(s_t)$ is the minimum cost of a sequence ending in state s_t, and $p(s_t)$ stores the s_{t-1} that results in this minimum.

5 Groundtruthing

(Details related to System Evaluation section of Methods)

There are two classes of errors made. Identity errors include flies swapping identity, flies that are lost, flies that are split into two detections, flies that are merged into a single detection, and detections that do not correspond to flies (Supplementary Note Sec. 5.1). Position errors are small, usually subpixel, errors in the estimated center position and orientation (Supplementary Note Sec. 6).

5.1 Identity errors

(Details related to Identity Errors section of Methods)

We evaluated the rate of identity errors made by our system on 18 video sequences (**Supplementary Table 1**). In each sequence, there were either 10, 20, or 50 flies. These flies were either all female, all male, or half male and half female. Two videos were evaluated for each condition. Each video was about 5 minutes long and recorded at 20 frames per second. For ease of counting, we began by putting 10 flies

[22] We used $\lambda = 0.05(px/fr)^{-2} = 1.12mm/s$ and $w_{max} = 0.25$.

in the arena then captured the 10 fly video. We then added 10 more flies and captured the 20 fly video. Finally, we added 30 more flies and captured the 50 fly video.

To detect identity errors made by the tracker, we would ideally watch each video multiple times in slow-motion. In each viewing, the video would be zoomed in on a different fly. As this would be extremely time consuming, we instead select for inspection frames and flies in which the tracker is more likely to make a mistake. We then watch these frames, zoomed in on the selected flies, in slow-motion. Tracking is easy when the flies are separated and their motion is well-predicted by the constant velocity model. It is more difficult when flies are close together or are jumping. For each video, we selected each frame and pair of flies that were touching (i.e. are part of the same connected component of foreground pixels, as described in Supplementary Note Sec. 4.4.2). We also selected all pairs of frames, flies in which swapping the identities of the pair of flies increased the error by less than a threshold. We set this threshold to 100. The units of this threshold is pixels squared, thus this is $(2.5\text{mm})^2$. In addition, we selected each frame and fly that did not move according to the assumed motion model (see Supplementary Note Sec. 4.5.1). In particular, we watched the frame and fly if the error of the center prediction was greater than a threshold (we chose 20 pixels = 5 mm). Finally, we watched the birth and death of each track.

Most of the frames watched contained no errors. We classified each error we observed into one of five types: lost fly, swapped identities, spurious track, merged flies, and split fly. **Supplementary Table 1** lists the number of each type of error observed in each video we evaluated. On average, we observed an identity error once every 5 fly-hours in the 10 fly videos, once every 1.5 fly-hours in the 20 fly videos, and once every 40 fly-minutes in the 50 fly videos. Example tracking errors are shown in **Figure 2d**.

Lost fly errors occur when the tracking system does not detect a fly in the arena. This will occur when both (1) the background subtraction step fails to classify enough pixels on the fly as foreground (see Supplementary Note Sec. 4.4.1) and (2) the hindsight step is not able to connect the two track pieces because the number of consecutive frames in which background subtraction fails is too large or the motion of the fly is too erratic during these frames. In our videos, all occurrences of this type of error were caused by errors in the background modeling when a fly spent more than half the video in the same position and became part of the background. The lack of motion of some of the flies only occurred in the 20 and 50 fly videos. In these videos, some of the flies had been in the arena for over 5 or 10 minutes, as discussed above. As the flies were in the arena longer, they seemed to grow accustomed to their environment and became less active. This suggests that it is best to collect video when the flies are first introduced to the arena and use this to estimate the background model. An explicit foreground model will also help prevent this type of error.

Swapped identity errors occur when the identity of two flies returned by the tracker swap at some point during the video sequence. This will occur when the motion model fails to predict the motion of at least one of the pair of flies, and the relative positions of the pair of flies is such that the error of swapping identities is smaller than not. We noted three situations in which this occurred. First, identities may be swapped if two flies jump at the same time in close proximity to one another. Second, consider a case in which one fly sits still and another fly jumps over it. As the tracker computes the error as the sum of squared distances for each matching of identity to observation, it prefers that each fly make a small jump rather than that one fly sit still and another fly make a large jump. This type of error can be prevented by modifying the hindsight step (Supplementary Note Sec. 4.6), and we plan to do this in the future. Third, a swap can occur if the splitting of a large connected component into multiple detections fails (see Supplementary Note Sec. 4.4.3). For instance, suppose one fly is actually just to the left of another fly. Occasionally, the clustering computed will actually split a connected component vertically rather than horizontally. In future work, we plan to address this issue as by incorporating a prior on the desired shape of the clusters into the clustering algorithm.

Spurious track errors occur when something that is not a fly is detected as a fly for a sequence of frames and is detected for enough frames so that the hindsight step does not correct the error. This type of error occurred twice in the videos. In the first case, a relatively large piece of debris in the arena at some point stuck to/was picked up by a fly walking over it. This piece of debris was therefore visible for less than half of the video and therefore was classified as foreground. In the second case, the spurious detection occured in the last frame of the video, thus it could not be resolved by the hindsight step.

We did not observe the other two types of errors. Before annotating the videos, we hypothesized the following definitions. *Merged* fly errors occur when two flies are detected as one for a sequence of frames long enough that the hindsight step cannot fix it. Similarly, *split* fly errors occur when a single fly is detected as two for a sequence of frames long enough that the hindsight step cannot fix it.

5.2 Fixing Identity Errors Manually

(Details related to Fixing Identity Errors Manually section of Methods)

It is possible to detect identity errors automatically and fix them manually in a matter of minutes. We found that some simple heuristics could be used to choose a small number of suspicious frames and flies. All of the labeled gross errors were in this set. With the tracker software, we also provide a GUI for automatically detecting suspicious frames and fixing the errors. A user can therefore observe this small set of suspicious frames and manually fix any observed gross errors with a minimal amount of work.

6 Position errors

(Details related to Position Errors section of Methods)

We developed two methods for evaluating errors made in estimating the position of each fly in two different ways. The first involved simultaneously recording a high resolution video of a portion of the arena and comparing the manually labeled fly positions in the high

resolution data to the positions returned by the tracker on the standard low resolution video of the entire arena. Supplementary Note Sec. 6.1 describes this evaluation. The second involved manually selecting tracks in which the fly looked to be moving with an approximately constant velocity. We then compared the tracks returned by the tracker to a very smoothed version of these tracks. Supplementary Note Section Supplementary Note Sec. 6.2 describes this evaluation. Both of these techniques resulted in qualitatively similar estimates of position error.

6.1 Comparing to High-Resolution Groundtruth

We captured 9 videos simultaneously with the original camera viewing the entire arena and with a close-focusing camera at 15x the original magnification, zoomed in on a small portion of the arena. **Supplementary Figure 1** shows frames of the same flies captured by each camera.

We manually labeled a random sample of 100 flies in the high-resolution video that were (1) fully visible and (2) not near another fly. For each sample, we drew the bounding box of the fly to estimate the center position and orientation of the fly. In addition, we also clicked on five points visible on each fly: the tip of the left antenna, the tip of the right antenna, the start of the left front leg, the start of the right front leg, and the scutellum. **Supplementary Figure 1** shows examples of the labels for some random samples.

To compare the high-resolution labels and the low-resolution tracks, we need to know the 2-D projective transform P taking coordinates in the high resolution video (x_h, y_h) to coordinates in the low-resolution video (x_l, y_l). This is a linear transformation with 8 degrees of freedom:

$$(cx_l, cy_l, c) = (x_h, y_h, 1)P$$

where P is an arbitrary 3×3 matrix with $P_{33} \neq 0$. We approximated the projective transform P using a few manually and automatically selected correspondence points on a grid. However, the projective transform computed in this way was inaccurate because (1) the number of correspondence points available was small and (2) the flies have some depth. Thus, we used the center positions of the labels and the center positions of the tracks to refine the projective transform. We found matches between flies in the low-res and high-res videos using the initial transform. Then, we updated P to minimize the error in predicting the center positions of the flies. Note that we used the same P matrix for all frames in all videos, as the two cameras did not move with respect to each other.

We used this same projective transform to compute error in orientation and labeled parts as well. As the tracker outputs the center position and orientation, not the locations of the predicted parts, for a given part we computed the average offset of the part along the major axis, normalized by major axis length, and the average offset of the part along the minor axis, normalized by the minor axis length. We use these values to predict the position of the part given just the tracked center position and orientation.

Supplementary Figure 2 shows the cumulative distribution of errors in estimating the center position, labeled part positions, and orientation. The median error in estimating the center is 0.0292 mm, equivalent to 2% of a fly's body length. The median error in estimating the orientaiton is 3.14°. The median error in estimating the part positions varies berween 0.10 and 0.16 mm.

We repeated the above experiment on 50 samples in which the chosen fly was close to another fly. Example labels for this test are shown in **Supplementary Figure 1b**. The mean and standard deviation of the errors are shown in **Supplementary Table 3**. As expected, the error when the flies are close is slightly larger.

6.2 Estimating Error from Fly "Wobble"

As it is difficult to setup a synchronized additional high-resolution camera and time-consuming to capture and label data from it, we evaluated a simpler, though possibly less accurate algorithm for estimating the amount of fine error. In a standard low-resolution video, we manually selected 10 sequences each of length 50 frames (2.5 seconds) in which the chosen fly appeared to be moving with an approximately constant velocity. As we believe in this sequence the fly's velocity is fairly constant, smoothing should not effect the true velocity too much. We smoothed 6 times with the filter $[1, 2, 1]/4$. We then computed the error between the raw track and the smoothed track for the center position and the orientation. The cumulative distribution of errors is plotted in **Supplementary Figure 2**. The distribution of center position error estimated from wobble during walks closesly matches the distributions of part location errors computed from the high-resolution video. The orientation error estimated from wobble during walks is smaller than that measured from the high resolution video. We hypothesize that this is because of variance in manual labeling the orientation. Still, these error measures are qualitatively similar to those computed from the high-resolution groundtruth data. Therefore, we find this second method appropriate for estimating position errors.

For comparison, we performed a similar evaluation using a fly that seemed to be static, following [9]. Again, we used 10 tracks of length 50 frames. Instead of smoothing the trajectory, we assumed that the position was static, thus computed the error to the mean center position and orientation. The errors estimated in this manner underestimte the true amount of error. We hypothesize that this is because pixelation effects do not occur when the fly is still.

References

[1] de Boer, P., Kroese, D., Mannor, S. & Rubinstein, R. A tutorial on the cross-entropy method. *Annals of Operations Research* **134**, 19–67 (2005).

[2] Straw, A. & Dickinson, M. Motmot, an open-source toolkit for realtime video acquisition and analysis. *Source Code for Biology and Medicine (under review)* (2009).

[3] Kimme, C., Ballard, D. & Sklansky, J. Finding circles by an array of accumulators. *Communications of the Association for Computing Machinery* **18** (1975).

[4] Reiser, M. & Dickinson, M. A modular display system for insect behavioral neuroscience. *Neuroscience Methods* **167**, 127–139 (2008).

[5] Piccardi, M. Background subtraction techniques: A review. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics* **4**, 3099–3104 (2004).

[6] Hastie, T., Tibshirani, R. & Friedman, J. *The Elements of Statistical Learning*. Springer Series in Statistics (Springer Verlag, Basel, 2001).

[7] Papadimitriou, C. & Steiglitz, K. *Combinatorial Optimization: Algorithms and Complexity* (Dover Publications, 1998).

[8] Cormen, T. *Introduction to Algorithms* (MIT Press, 2001).

[9] Valente, D., Golani, I. & Mitra, P. Analysis of the trajectory of *Drosophila melanogaster* in a circular open field arena. *PLoS ONE* **2** (2007).

Nature Methods: doi:10.1038/nmeth.1328